

AI-Driven Dynamic Reorder Point Prediction for Inventory Management in Supply Chains: A Machine Learning Approach

SK Ayub Al Wahid¹, Tapan Chandra Ghos², Ashadujjaman Sajal³, Anni Akter⁴, Abul Hasnat Md Faysal⁵, Adnan Hasan⁶, Nagib Mahafuz Turzo⁷, Pravakar Debnath⁸, Nisha Gurung⁹ and Reza E Rabbi Shawon¹⁰

¹Doctor of Management (DM), International American University

²School of Engineering & Technology, Western Illinois University, Macomb, Illinois, USA

³Management Science & Quantitative Methods, Gannon University, Erie, PA, USA

⁴Department of Business Administration and Management, International American University

⁵MS in Industrial and Systems Engineering, University of Buffalo, NY, USA

⁶MS in Industrial Engineering, University of Buffalo, NY, USA

⁷Bachelor of Business Administration, International American University

⁸School of Business, Westcliff University, Irvine, California, USA

⁹MBA Business Analytics, Gannon University, Erie, PA, USA

¹⁰MBA Business Analytics, Gannon University, Erie, PA, USA

Corresponding author: SK Ayub Al Wahid, Email: alwahid.ayub@gmail.com

HOW TO CITE:

SK Ayub Al Wahid, Tapan Chandra Ghos, Ashadujjaman Sajal, Anni Akter, Abul Hasnat Md Faysal, Adnan Hasan, Nagib Mahafuz Turzo, Pravakar Debnath, Nisha Gurung, Reza E Rabbi Shawon (2026). AI-Driven Dynamic Reorder Point Prediction for Inventory Management in Supply Chains: A Machine Learning Approach. International Journal of Special Education, 41(4s), 441-461.

ABSTRACT:

Keeping the right amount of stock in a modern supply chain is a tricky balancing act. It comes down to making sure products are there when people want them without spending a fortune on warehouse space, all while demand shifts and changes in ways that are hard to pin down. Most places still use old-school reorder point methods. These work fine enough, but they usually bank on everything staying the same, which just isn't how retail works anymore. This research looks into how AI might handle those shifting numbers better by creating dynamic reorder points based on actual retail transaction data. It involved building a specific pipeline to turn messy sales records into clear demand patterns for individual items, then testing out various machine learning models to see if they could actually predict what happens next. Those predictions were then plugged into a system designed to adjust reorder levels on the fly and tested in a simulated environment. The numbers show that machine learning does a much better job of guessing demand than the basic statistical shortcuts people used to rely on. When these models were used to set reorder points, the system got much better at preventing empty shelves and keeping customers happy. But there is a catch. Better service usually meant holding onto more stock, which points to a clear tension between being reliable and being cheap. Looking closer at the data, it became obvious that how long it takes for a shipment to arrive really changes how well these policies work. Plus, even the smartest models struggled when demand went totally off the rails or when weird, one-off transactions popped up. In the end, it seems that using Machine Learning

COPYRIGHT STATEMENT:

Copyright: © 2026 Authors.
Open access publication under the
terms and conditions
of the Creative Commons Attribution
(CC BY)
license (<http://creativecommons.org/licenses/by/4.0/>).

to move reorder points around can actually help a business run better, provided the system is set up thoughtfully. This work offers a clear, repeatable way to link demand guesses with inventory simulations, giving a solid starting point for anyone trying to build better decision tools for supply chains. It also serves as a reminder that a more accurate forecast only matters if it actually leads to a smarter decision on the warehouse floor.

Keywords: Dynamic Reorder Point; Inventory Management; Machine Learning; Supply Chain Analytics; Demand Forecasting; Stockout Reduction; Safety Stock; Retail Transactions.

1. INTRODUCTION

1.1 Background and Motivation

Deciding when to order more stock is the heartbeat of running a warehouse. It boils down to a single question: when is the right time to trigger a new shipment before the shelves go empty? This is not just a minor chore for a manager to handle. It touches everything from whether a customer leaves happy to how much cash is stuck sitting in a backroom. If the trigger happens too late, the store runs out, sales are lost, and shipping gets expensive because everyone is panicking. If it happens too early, the warehouse gets cluttered, items might expire or go out of style, and money is wasted on products that are just gathering dust. This tension is tough in retail because every product has its own personality. Some sell like clockwork, while others sit still for months before disappearing in a sudden rush. It was noted by Chopra and Meindl (2016) that good planning requires a balance between being fast to respond and staying cheap, which makes the reorder strategy a huge part of how well a business actually functions [9].

Old-school ways of picking a reorder point are still popular because they make sense. It involves looking at how much people usually buy while waiting for a delivery, adding a little extra "just in case" stock, and hitting the order button when inventory hits that mark. It is a simple and clear method. But it only works if the numbers put into the formula are actually good. In the real world, demand is rarely flat or predictable. People buy differently depending on the week, the season, or even the price. A fixed number that worked in the summer might be totally wrong by December. This

gets even messier when a company has thousands of different items to track, each with its own weird sales patterns. Silver, Pyke, and Thomas (1998) looked at inventory as a constant puzzle of timing and uncertainty, suggesting these trigger points should be treated more like flexible plans rather than permanent rules [26].

This research came about because there is a big gap between those textbook formulas and the messy reality of daily sales. Many stores have mountains of data, but still use basic averages or manual guesses to decide when to restock. This is where machine learning might actually help. The point is not that a computer model can magically fix a supply chain. A better way to look at it is that these models might be better at the "guessing" part of the formula. If a model can spot patterns in past sales, price changes, or seasonal dips, the reorder point it suggests might actually fit the situation. The idea is to use technology as a support tool, not a total replacement for human judgment. At the end of the day, a model is only good if it helps the business, not just because it has a high math score. The goal here is to see if these high-tech, moving reorder points actually do a better job than the old-fashioned, static ones.

1.2 Problem Statement

A lot of inventory systems are still stuck using old numbers or slow-moving averages. That might be fine if life is predictable and you only sell a few things, but it falls apart when demand starts jumping around. In retail, what people want changes by the day and by the product. Using one rigid rule for everything leads to a mess. Fast-selling items run out too often, and slow-moving

stuff piles up until there is no room left. The real headache is not just predicting the future; it is turning those predictions into a specific plan that keeps the shelves full without overspending. Carbonneau, Laframboise, and Vahidov (2008) showed that various types of neural networks and machines can be used for these forecasts, often beating out basic averages when things get chaotic in the supply chain [8]. The core issue explored here is whether forecasts from machine learning can be turned into restocking triggers that actually work better than the basics. It was important to look past just the "accuracy" of the math. If a paper only talks about error rates, it misses the point of managing a business. These methods have to be judged on what happens in the warehouse, things like how often the shelf is empty, how much stock is sitting around, and how much it all costs. Waller and Fawcett (2013) pointed out that data science is most useful when it is tied directly to a real decision rather than being a math project in a vacuum [28].

There is a catch that needs to be mentioned. The dataset used, which comes from an online retailer, shows what people bought but does not show the actual warehouse levels or when the company sent out purchase orders. Because that info is missing, it was impossible to see exactly what the original company did. Instead, the sales data was used to build a history for each product, and then a simulated inventory system was built on top of it. The models predict the demand; those predictions are turned into reorder points, and then the whole thing is tested to see how it would have played out. It is a solid way to test things, as long as the setup is clear. The main problem being solved is that too many studies focus on the math of the prediction and ignore the actual move that follows. This study connects those dots by linking the forecast to the restocking trigger and then testing it in a simulated store. It is not assumed that AI is always the winner. Instead, it was tested to see if it actually provides a clear win under specific conditions. Success here is defined simply: the new way has to keep the shelves stocked without letting the inventory levels get out of hand.

1.3 Research Objectives

The main goal here is to figure out if using machine learning to set dynamic reorder points actually stops shelves from going empty without piling up too much extra dust-collecting stock. Instead of just making a fancy demand forecast and calling it a day, this work uses those forecasts as a starting line for making real inventory choices. In a more hands-on sense, the models guess how much of a specific item will sell, and then that guess gets turned into a reorder trigger using a safety stock formula. These triggers then get tossed into a simulated warehouse setup to see how they hold up. This approach gets to the heart of a much better question: it is not just about whether the math predicted the sales numbers right, but whether those numbers actually led to better choices about when to buy more stock. There are a few other things to check on the side. First, it is worth looking at how these forecasts handle different types of products. Some items fly off the shelves, while others barely move, and a model that looks good on paper might actually be a disaster for those slow-moving or unpredictable items. Second, this study puts old-school static rules, simple rolling averages, and new machine learning methods into the same simulation. It is a fair fight that way since every method has to play by the same rules. Third, the work tests how things change if it takes longer for an order to arrive. Since the data doesn't explicitly say how long shipping takes, a sensitivity analysis checks if the models still work when those wait times shift around. Finally, there is a deep dive into the mess-ups, looking specifically at where models guessed way too high or way too low and why that created a weak spot in the operation.

These goals stay grounded in reality. There is no attempt to prove that machine learning is a magic fix for everything in inventory theory. Instead, the focus is on whether it can help one specific part of the process under certain conditions. Inventory is always a balancing act. A strategy that fixes stockouts might only do so by ordering way too much stuff, and a lean approach might leave customers frustrated with empty shelves. A system is only as good as its ability to find the middle

ground. Because of that, the evaluation looks at both the accuracy of the guess and the health of the inventory, making sure the improvements are real and not just a statistical fluke.

1.4 Research Contributions

This work adds a few useful pieces to the puzzle of how machine learning fits into supply chain tasks. To start, it builds a repeatable testing pipeline using real retail sales data. Rather than playing it safe with fake, perfectly clean data, this uses actual transaction records that show all the weird, messy gaps found in real stores. This makes the whole setup feel much more like the real world, yet it is still easy for others to pick up and run in a standard research notebook. The process covers everything from cleaning up the data and picking the right features to training the models and running the final simulations. Another contribution is the way the dynamic reorder points are actually built. This paper doesn't try to force the model to predict a reorder point directly, which would be tough since there are no "correct" labels for that in the data. Instead, the machine learning handles the demand side, and then standard inventory logic takes over to set the thresholds. It's a more logical way to work that respects both the data and the way a warehouse actually functions. Along with that, a custom simulation engine was built to test these policies. This means performance is judged on things like fill rates and total costs rather than just looking at error bars on a graph.

The study also puts classic inventory logic right next to modern machine learning in a controlled head-to-head match. By comparing static rules and rolling averages against things like Ridge Regression, Random Forests, and XGBoost, it becomes clear if the extra complexity is actually worth the trouble. One of the biggest takeaways is the dual-evaluation method. It connects the accuracy of the forecast directly to the outcome on the shelf. This matters because, in the end, an inventory decision shouldn't be judged by how close the prediction was, but by whether the right amount of product ended up in the right place. Lastly, the addition of sensitivity and error analysis makes the findings a lot more sturdy. Testing

different lead times ensures the results don't fall apart if shipping slows down, and breaking down errors by demand type shows exactly where the models shine or stumble. All these pieces together turn this from a simple coding exercise into a proper experiment for making better decisions about when to restock.

1.5 Scope and Assumptions

This work stays focused on making weekly inventory decisions at the individual SKU level. The idea was to take retail transaction data, turn it into demand forecasts, and then flip those forecasts into dynamic reorder points. To see if any of this actually worked, everything was run through a simulated replenishment setup. It is important to note that this is not a full-scale enterprise inventory system. There was also no attempt to guess or recreate what the retailer actually did in real life. The data just wasn't there for that; it was good enough for modeling demand, but not for seeing exactly what was sitting on the shelves at every moment. The simulation itself was built on a few specific rules: a weekly review cycle, lead times that don't change, and a "lost sales" setup where a customer just leaves if the item isn't there. It also used simplified costs for holding stock, running out of items, and placing orders. Bertsimas and Thiele (2006) once talked about using robust optimization to handle inventory when nobody knows the exact demand distribution [6]. That kind of thinking about uncertainty matters here, even though this study leans on machine learning and simulation instead of its specific math.

It was also assumed that whatever sales showed up in the data could count as demand. Everyone knows that's a bit of a stretch because if a store runs out of milk, you can't see the people who would have bought it. That is just part of the deal when working with transaction logs without extra stockout records. The costs used in the simulation aren't the retailer's real bank statements either. They are just consistent numbers used to see how different strategies stack up against each other. Islam et al. (n.d.) looked at AI as a bridge between making predictions and actually managing things like working capital or keeping customers happy

[14]. That fits right in to use inventory thresholds as a lever to balance having enough stock without burying the business in extra boxes. Babai, Ali, Nikolopoulos, and Boylan (2012) also pointed out something pretty vital: a better forecast doesn't always mean a better inventory result [4]. They are related, sure, but they aren't the same thing, which is why it was necessary to look at the actual inventory outcomes rather than just getting excited about a high-accuracy forecast.

2. LITERATURE REVIEW

2.1 Classical Inventory Control and Reorder Point Models

The whole backbone of supply chain management comes from classical inventory theory. It's basically just a big balancing act. On one side, there is the cost of keeping stuff in a warehouse, and on the other, the risk of having to tell a customer "no." Most of these early ideas assume demand is a bit random and lead times are mostly fixed. The reorder point model is the go-to because it is easy to understand and actually use. The rule is simple: when the stock level hits a certain line, buy more. That line is usually the amount expected to be sold before the new shipment arrives, plus a little extra "safety" cushion. That cushion depends on how much the demand jumps around and how many customers the store is willing to potentially disappoint. Hadley and Whitin (1963) were some of the first to really sit down and map out the math for all this [12]. They looked at systems that check stock levels constantly and those that only check once in a while, creating the blueprint for how demand, costs, and timing all play together.

Later on, other people started building onto those basics to make them fit the messy real world. Axsäter (2006) took these models and applied them to complicated chains with multiple locations and lead times that aren't always the same [3]. He showed that reorder points can work even in tough spots, but the whole thing hinges on how well you understand the variability of the supply and the demand. In the real world, people often take shortcuts, like assuming demand never changes or that the truck always arrives exactly on time. That doesn't always work out in a real store. There's often a big gap between what looks good on a

chalkboard and what happens on a shelf. Classical models are great because they give clear answers, but they can be a bit stiff when demand starts moving fast or when every product behaves differently. Static reorder points based on old averages get stale if there is a holiday or a sudden shift in what people want. This is why the literature keeps pointing toward the need for more flexible ways to estimate demand, especially now that there is so much data available to help.

2.2 Demand Forecasting for Inventory Management

Getting inventory right usually comes down to how well someone can guess what customers will want later. Because of that, demand forecasting is basically the heart of the whole operation. For a long time, the standard way to do this involved time series models, things like moving averages or ARIMA. These are popular because they aren't "black boxes"; it is easy to see how they work, they aren't hard to set up, and they do a solid job if things stay relatively calm. A good look at this comes from Hyndman and Athanasopoulos (2018), who pointed out that there isn't one "best" way to forecast everything [14]. It really depends on what the data looks like, whether there's a big trend, if things change with the seasons, or if the data is just messy and noisy. They argued that checking for errors and making sure the model actually fits the reality of the data is the only way to make sure these guesses are actually useful for making inventory choices.

Recent tests on a massive scale have backed up the idea that picking the right tool for the specific job is what matters most. The M4 competition, run by Makridakis, Spiliotis, and Assimakopoulos (2018), put a ton of different methods head-to-head, from old-school math to newer machine learning [29]. A big takeaway was that mixing different methods usually worked better than just picking one. Also, surprisingly, the simpler methods still held their own quite well. The results showed that demand can look totally different from one item to the next, so a system has to be flexible enough to handle that variety. While everyone is talking about machine learning because it can spot weird, non-linear patterns, those classic statistical methods are

still very reliable, especially when there isn't a mountain of data to work with.

The real trick with inventory isn't just getting a number that looks right on a spreadsheet; it's making sure that number helps run the business. Being wrong has different costs depending on which way the error goes. If the guess is too low, the shelves go empty, and sales are lost. If it's too high, money gets tied up in stuff sitting in a warehouse. Because of that, just looking at standard math errors like mean squared error doesn't tell the whole story. A forecast has to be judged by how it actually affects things like stockouts or overstock. Most of the research suggests that connecting these forecasts to actual inventory rules takes a lot of care, especially when looking at how long it takes to get new stock and how much uncertainty there is. There is still a lot of room to figure out how to weave these predictions into day-to-day policies, especially when a company is dealing with thousands of different products and messy transaction logs.

2.3 Machine Learning in Supply Chain Inventory Optimization

Now that companies can track every single transaction, there is a lot of excitement about using machine learning to fix supply chain headaches. These models are great at finding complicated patterns that older math models might miss entirely. It might be the way a price drop interacts with a specific holiday, or how certain types of shoppers behave at different times of the year. Kourentzes, Petropoulos, and Trapero (2020) talked about how important it is to break a time series down into pieces like trends and seasonal shifts [18]. They showed that getting better at identifying these pieces leads to much better forecasts. Machine learning is naturally good at this, especially if the person building the model feeds it the right info, like past sales, rolling averages, or even just the calendar. One of the best things about machine learning is how it deals with a world that won't stay still. In real life, demand isn't a straight line; it jumps around as people change their minds or new trends pop up. Bhowmik et al. looked at models that can actually teach themselves to adapt when the data starts acting

differently [7]. While they were looking at finance, the same logic applies to a warehouse. Since things like big promotions or sudden shifts in what's "cool" can totally change demand overnight, having a model that learns on the fly gives it a huge leg up over a static formula that never changes.

Still, it isn't a magic fix. A major hurdle is that a "more accurate" forecast doesn't always mean the inventory situation gets better. A model might be technically very accurate on average, but still fail miserably by underestimating demand right when a big rush happens. That leads to empty shelves, which is the one thing everyone wants to avoid. These models also take a lot of work to set up, you have to pick the right features and tweak the settings constantly so the model doesn't just memorize the past and fail to predict the future. The consensus in the field is that while machine learning can definitely make better guesses, it has to be built with the actual goals of the business in mind to really be worth the effort.

2.4 Simulation-Based Evaluation of Inventory Policies

Trying to figure out if an inventory policy actually works means looking at how demand, restocking choices, and shelf levels dance together over time. It is a bit of a puzzle because, in the real world, clean data on what was actually on the shelf or how many people walked away empty-handed is often missing. It is surprisingly hard to pull those details out of messy transaction logs. Because of this, the simulation ends up being a great workaround. It lets people build a controlled version of an inventory system to see how a strategy holds up under different conditions. Discrete-event simulation, as noted by Banks, Carson, Nelson, and Nicol (2010), works by tracking a system where things only change at specific moments, like when a customer buys something or a truck arrives with more stock [5]. This makes it a perfect fit for inventory, where the whole story is just a series of events like running out of items, placing an order, and waiting for it to show up.

The real beauty of using simulation is that it shifts the focus toward how things actually run on the ground, rather than just looking at math scores.

Law (2014) pointed out that these models are great for estimating things people actually care about, like how much work gets done, wait times, or total spending [19]. For a warehouse or a store, that translates to the fill rate, how often things go out of stock, and how much money is tied up in sitting inventory. These numbers tell a much better story than a forecast error score. A plan might be a little off on its predictions, but it still keeps the shelves full and the costs low, which is what really matters in the end. There is also a lot of freedom in simulation to play with "what if" scenarios. It is easy to tweak lead times, mess with demand swings, or change cost goals to see when a policy might break. This is a huge help when the original data is a bit thin, as the simulation can fill in the blanks to create a fair testing ground. Of course, the results are only as good as the guesses put into the model. If the starting assumptions are off, the whole thing falls apart, so those choices have to be explained clearly. Ultimately, the literature treats simulation as the bridge that connects abstract predictions to the messy reality of running a business, especially when the available data isn't perfect.

2.5 Research Gap

Even with all the work done on forecasting and inventory, these two worlds often stay in their own lanes. A lot of papers spend all their time trying to make a forecast slightly more accurate without ever checking if that actually helps a manager make a better order. On the flip side, old-school inventory math often assumes the demand is just a known, steady thing, totally ignoring the powerful data tools available today. Fildes, Goodwin, Lawrence, and Nikolopoulos (2008) mentioned that people often have to step in and tweak statistical forecasts using their own judgment to make them work in practice [11]. This just goes to show that numbers on a page don't always capture the whole truth of a retail environment. This disconnect is even more obvious when looking at individual store sales. Demand for a single item can be all over the place and depends on a dozen different factors. It is rare to find research that joins the dots all the way from predicting demand for a specific item to creating a moving reorder point and then testing it all with a simulation. Without that full loop, a lot of

forecasting models just aren't that useful for a real supply chain. There also isn't much out there in terms of a start-to-finish process that handles the data cleaning, the machine learning, and the simulation in one go. This study tries to fill that void by tying forecasting and inventory decisions together in one big experiment. The goal is to see if better predictions actually lead to smarter choices, adding something useful to both sides of the conversation.

3. METHODOLOGY

3.1 Research Design

It was decided to set up the study as a simulation experiment driven by transaction demand. The goal was to see if using machine learning to set dynamic reorder points actually works better than the old-school ways of managing inventory. It is worth noting that predicting a reorder point isn't handled here as a simple math problem where the answer is already known. Instead, it was built as a layered process where forecasting how much people will buy is just the first step in deciding when to buy more. This matters because the data doesn't actually show when the store ordered more stock in the past. The focus stayed on making a path from looking at sales to seeing how those sales would play out in a simulated warehouse.

The whole process starts with messy retail sales data. It goes through a few rounds of cleaning, grouping items by the week, filtering out products that don't fit, and picking out the right features. After that, the time periods are split up to keep the testing fair. This cleaned-up info goes into both basic stats models and newer machine learning ones to guess future demand. Those guesses are turned into reorder points by adding a bit of a safety cushion based on how often the store wants to stay in stock. These numbers are then dropped into a simulation that acts like a real-world inventory system. It tracks things like how often items ran out or how much money was tied up in extra stock, which makes it easier to tell which method actually performed best. This setup was chosen so the results would show the outcome of the whole decision, not just whether a forecast was a little bit off. It links together the guessing, the ordering, and the final results. It is based on the

idea that an inventory plan is only as good as what happens on the shelves. There was also some extra testing done to see what happens if shipping takes longer than expected or if the models struggle with certain types of products. Doing it this way makes sure the findings are about real-world results rather than just numbers on a spreadsheet.

3.2 Dataset Description

The actual numbers used for this come from the UCI Online Retail II collection. It is basically a big pile of sales records from a shop in the UK over a couple of years. Because it lists every single thing someone bought, it is perfect for building a timeline of demand for each product. Every row in the data is a line on an invoice. It has the product code, when it was bought, how many were sold, the price, and a bit about the customer and where they live. The "StockCode" is what identifies the specific item, and the "InvoiceDate" lets the study track how sales go up and down over time. There are some quirks in the data, like negative numbers. Usually, those mean someone returned an item or canceled their order. This makes things a bit tricky because what shows up as a sale might not be the "true" demand if lots of people are sending stuff back. There are also some spots where information is missing, especially the customer IDs, so that had to be handled carefully. While the shop sells to different countries, the study mostly looks at the total sales for each item rather than worrying about where the buyer was located.

One big thing to keep in mind is that the data says nothing about the warehouse itself. There is no info on how much was sitting on a shelf, when the manager placed an order, or how long a supplier took to deliver. Because of this, it is impossible to know exactly what the retailer was doing at the time. The data is used to build a model of what people wanted to buy, and that model then feeds into a simulated version of a supply chain. It acts as a stand-in for real demand, which is a fair way to test the logic without pretending the data shows the full picture of the business operations.

3.3 Data Ingestion

The way the data is brought in was kept simple so anyone else could do it too. It starts by pulling in

the usual Python tools for moving data around and training models. The dataset is grabbed straight from the UCI website using a link, so there is no need to manually download or move files around. After it is pulled in, the data is saved in a standard way so the experiment stays consistent every time it runs. This helps avoid problems if an external link changes or goes down later. During this part, some quick checks are run to make sure the file isn't broken, the columns are named right, and the numbers look like numbers. A few rows are poked at just to be sure things like dates and prices didn't get scrambled during the upload. The "InvoiceDate" column gets formatted into a proper date format so it can be used to group sales by week or month. If a date looks weird and can't be read, it gets marked as empty so it can be fixed later in the cleaning phase. Doing these checks early keeps mistakes from hiding in the code and popping up later when the models are running. It creates a solid starting point for all the work that follows.

3.4 Raw Data Audit

Before any actual changes happen, the raw dataset gets a thorough walkthrough to see what's really there. It's about figuring out the structure and finding where things might be messy before the math starts. Jumping straight in without looking is a good way to end up with weird results or models that don't make sense later on. Things start simply: counting up all the rows and columns to get a feel for how big the pile of data actually is. It also matters whether the columns are numbers or categories, so the data types get checked to make sure everything is sitting where it should be. The timeframe is another big piece of the puzzle. By looking at the earliest and latest dates in the InvoiceDate column, the boundaries of the study become clear. Knowing exactly how many months or years are covered helps decide if there's enough history for time-series work. Then there's the issue of what's missing. Every column gets checked for empty cells, with a real focus on the heavy hitters like StockCode, Quantity, and Price. If those are blank, building a demand model becomes pretty much impossible.

This part of the process also looks at how much variety is in the mix. It counts up how many unique products and customers are involved, which shows how diverse the business activity really is. This helps later when deciding which items to keep or toss. Then there are the numbers themselves. The audit looks for extremes in quantities, especially those negative numbers that usually mean a return or a canceled order. Prices get the same treatment to spot zeros or negatives that might be typos or perhaps just freebies. Cancellations get their own specific check by looking for certain prefixes in the invoice codes. It's a way to separate a regular sale from a transaction that got reversed. Doing it this way means the data isn't just accepted at face value; it's poked and prodded for inconsistencies. This whole audit keeps things transparent and makes sure the next steps are based on what the data actually is, not what it's supposed to be.

3.5 Data Cleaning Protocol

The goal of cleaning is to turn that messy raw data into something a model can actually use without losing the interesting bits about sales and returns. First, the trash goes out: any records missing the basics, like what the product is, when it was sold, how many, or for how much, are just removed. Without those details, a record doesn't really add anything to a demand series. The InvoiceDate also gets turned into a standard format so the computer reads the timing correctly, and any dates that don't make sense are cut. Prices that are zero or less are also taken out since those are usually errors or weird edge cases that don't fit a standard sales model. Quantities get a closer look, too. If a quantity is zero, it's gone because it didn't change anything. But negative quantities stay in the mix for a bit. Instead of just deleting them, they are used to figure out the difference between net and gross demand. Net demand is the total after subtracting returns, while gross demand looks at the total volume using absolute numbers. Keeping both versions gives a much better picture of how people are actually buying and returning things.

A few more steps help tidy things up, like making sure all the product IDs look the same. Revenue is calculated by multiplying the quantity by the price,

which lets the study look at money and units at the same time. The timestamps also get broken down into years and weeks. This makes it easier to stack the data into weekly buckets for each product, which is exactly what's needed for the inventory simulations coming up later. The whole cleaning process follows a set of rules applied to everything, so it's easy to see what changed and why. By dealing with missing info and weird entries in a logical way, the final dataset stays solid. Most importantly, it keeps the real-world chaos of returns and demand swings intact, which is the only way to get an inventory model that actually works in a real warehouse.

3.6 Demand Construction, Aggregation, and Feature Engineering

Turning messy transaction logs into something useful for inventory modeling meant organizing everything by SKU and week. Each product, identified by its StockCode, was grouped into weekly chunks to create a clear timeline of demand. Picking a weekly window felt right; daily data is often too patchy and full of holes, while monthly data tends to smooth over the quick jumps and drops that actually trigger a reorder. This middle ground keeps the timing details sharp without getting lost in the noise of empty days. For every SKU-week, several variables were pulled together to describe what happened. There is net demand, which is the total quantity after subtracting returns, and gross demand, which looks at the total volume of all transactions combined. Other details were tracked too, like how many invoices were sent, the average price, total money made, how many different customers showed up, and the return rate. Time markers like the specific year-week and the start date of that week were kept to make sure everything stayed in the right chronological order.

Then came the feature engineering to help the models understand how demand moves over time. Lag features were built to show what happened one, two, three, or four weeks back. Rolling stats, like the average, standard deviation, and how much the demand swung around, were calculated over four and eight-week periods to catch recent trends or sudden volatility. Highs and lows from those same windows were also included to show the

range of recent activity. Beyond just demand numbers, the models got to see things like price history, customer counts, and return ratios to get a better handle on what drives buying habits. Calendar bits like the month, quarter, and a flag for December were added to account for the holiday rush and other seasonal cycles. A simple counter for how many weeks a product had been around was used to track its lifecycle. A major rule here was making sure the model never got a "sneak peek" at the future. Every lag and rolling stat was built strictly from past data. To keep things honest, rolling windows were shifted so that whatever was happening at time t was only based on what occurred before t . This setup mimics a real-life situation where nobody knows next week's sales yet. It is a vital step because it keeps the results realistic and stops the model from looking better on paper than it would actually perform in a warehouse.

3.7 Data Filtering, Target Definition, and Temporal Splitting

Retail data is a mixed bag, and not every product is a good candidate for this kind of modeling. Some items sell so rarely or so randomly that they just create confusion. Because of that, a filtering process was used to keep only the products with enough life in them. Only SKUs with a decent number of active weeks, a minimum amount of total sales, and at least a few weeks of positive demand were kept. This keeps the focus on products where there is enough of a pattern to actually learn something. Now, cutting out the slow movers could make the results look better than they really are by focusing only on "easy" products. To stay transparent, the total count of SKUs before and after this cleanup is documented, along with the retention rates and the stats for what was left. Products were also sorted into low, medium, and high-volume buckets. This way, it is possible to see how well the models work across different levels of popularity.

The goal for the model is to predict what happens next, not what is happening right now. Specifically, the features from time t are used to guess the demand at $t + 1$. This keeps the forecasting grounded in reality. It is also possible to look

further ahead by totaling up demand over a few future weeks, which helps account for the time it takes for a new shipment to actually arrive. The data was chopped into training, validation, and testing sets using a hard line in time. The oldest data was used for training, the middle chunk for tuning, and the most recent slice for the final test. This chronological split is much better than picking random days because it forces the model to predict a "future" it has never seen, which is exactly how it would be used in a real business.

3.8 Forecasting Models and Baseline Methods

To see if fancy machine learning is actually worth the effort, it had to go up against some old-school methods. These baselines included basic reorder point math, rolling statistical targets, and "naive" guesses. A static reorder point uses the long-term average and a safety buffer based on how much demand usually fluctuates. A rolling version of that does the same thing but updates as new data comes in, allowing it to drift with the trends. Simple averages from the most recent weeks provided another basic bar to clear. Then the more complex models were brought in. A Ridge Regression model was used to check for straightforward, linear patterns. For the more complicated, messy relationships in the data, Random Forest and XGBoost were used. These were fed the whole library of engineered features and checked against the validation data to make sure they weren't just memorizing the past. The settings for these models, the hyperparameters, were picked based on what worked best on the validation set. Everyone was treated the same here, and the test set was kept totally off-limits during this phase to avoid any bias. This ensures that if one model beats another, it is because it's actually better at forecasting, not just because the settings were fiddled with to fit the final test. The point wasn't just to find the most accurate number, but to see if using a smarter model actually leads to better decisions about when to order more stock.

3.9 Dynamic Reorder Point Formulation and Inventory Simulation

The raw numbers coming out of the forecasts get turned into actual reorder points by leaning on a safety stock framework that cares about service

levels. For every single item and time slot, the reorder point is just the expected demand during the wait time plus a little extra padding to handle uncertainty. This buffer is usually a mix of the desired service level and how much the forecast tends to wiggle. A basic way to do this is just looking at the standard deviation of the guesses, but it works better to look at the leftover variability in the residuals. Doing it that way does a better job of catching the real gaps in the demand estimates. These changing reorder points get put through the wringer in a simulated inventory setup that acts like a standard periodic review. Everything happens week by week. It is assumed that the lead time stays the same, and if a customer doesn't find what they need, the sale is just lost since there are no backorders here. The system uses an order-up-to approach. To figure out where things stand, the inventory position is tallied up by adding what is on the shelf to what is already on the way. If that total number drops below the reorder point, a new order gets triggered.

To keep things grounded in reality, the starting stock levels aren't just some random flat number. Instead, they are set based on how that specific item actually behaves. The amount ordered is calculated to bring the inventory back up to a set target above the reorder point. This keeps the logic consistent with standard inventory theory instead of just making up rules on the fly. Throughout the process, the simulation keeps a close eye on stock levels, when orders go out, how often shelves go empty, and the total bill. That bill is a combination of holding fees, penalties for stockouts, and the cost of placing the orders. These price points are mostly for show, simplified versions used to compare different strategies rather than actual accounting data. Using this kind of framework helps turn abstract forecast accuracy into actual operational results, which is a much more useful way to see if a model actually works.

3.10 Evaluation Framework and Robustness Analysis

The success of the model is measured by looking at both the forecast numbers and how the inventory actually holds up. This gives a full picture of how these dynamic policies play out. On

the forecasting side, things like mean absolute error and root mean squared error are used. It helps to use weighted metrics rather than just standard percentage errors, mostly because those percentages get really messy and unstable when demand is close to zero. On the operational side, the focus shifts to things like fill rates, the total number of times stock ran out, average shelf levels, and how often orders were placed. These are the things that actually matter in a warehouse, since the goal is to balance keeping customers happy with keeping costs down. Using both types of measurements ensures the models aren't just good at guessing numbers but are actually helpful for making decisions.

To make sure the findings aren't a fluke, a sensitivity analysis is run by messing with variables like lead time. Testing different delays shows how the model holds up when the wait gets longer or shorter. Other tests look at how things change across different types of items or when the target service level is shifted around. An error analysis is also tucked in there to find the weak spots, like when demand is super low, when it suddenly spikes, or when things just get generally chaotic. Everything is kept reproducible by sticking to the same data cleaning steps, locking in random seeds, and following a set routine for every experiment. All the results are saved and written down so the work can be checked or built upon later. This whole setup is designed to make sure the conclusions are solid, easy to understand, and actually applicable to real-world inventory problems.

4. RESULTS AND EVALUATION

The findings from this study are laid out in a few different layers, covering the whole journey from checking data quality to seeing how the inventory actually holds up. It wasn't just about whether the models could predict numbers accurately; the goal was to see how these forecasts actually played out in a simulated warehouse setting. This gives a much clearer picture of how dynamic reorder points handle the messy reality of daily operations.

4.1 Data Audit Results

The first look at the data shows it is both massive and a bit chaotic. There are 525,461 transaction records here, covering 373 days and 40 different countries. That is a lot of variety in how people shop. But with that scale comes some baggage. About 1.94% of the entries are just cancellations, and the Customer ID field is blank for more than a fifth of the records. There are also some weird

spots, like 12,326 instances of negative quantities and 3,690 items with prices that aren't positive. This points to returns, mistakes, or just messy bookkeeping. After a thorough cleaning, 99.3% of the records stayed intact. Keeping that much data was vital because cutting too deep might have wiped out the very demand patterns the study aimed to find, yet leaving the errors in would have just created a bunch of noise.

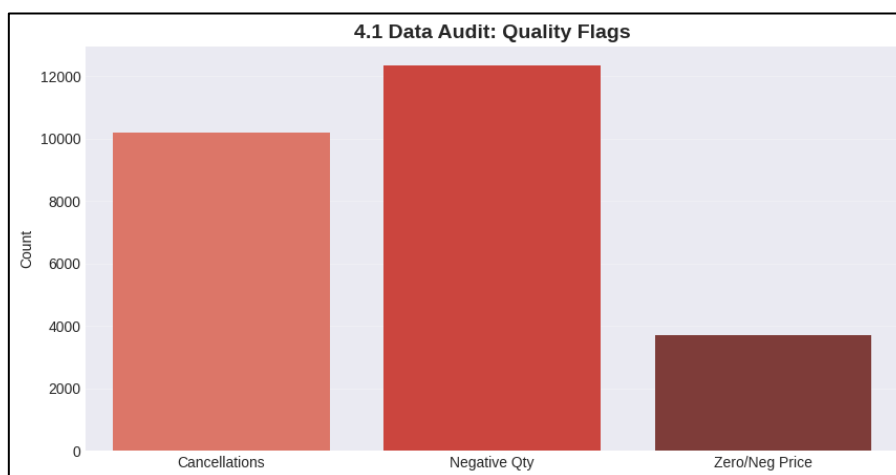


Fig.1: Data Audit Quality Flags

4.2 SKU Selection Results

Once the data was cleaned up, it was time to pick which products to actually track. Out of 4,115 unique items, 2,206 made the cut based on how often they were sold. That is a retention rate of 53.6%. This ensured the models had enough history to actually learn something. These items

were split pretty evenly between low, medium, and high-volume sellers. On average, these products moved about 53 units a week, but the coefficient of variation was high at 1.39. That is a lot of swinging back and forth. With demand being that jumpy, it makes sense why a fixed, "set it and forget it" reorder point wouldn't really work well across the board.

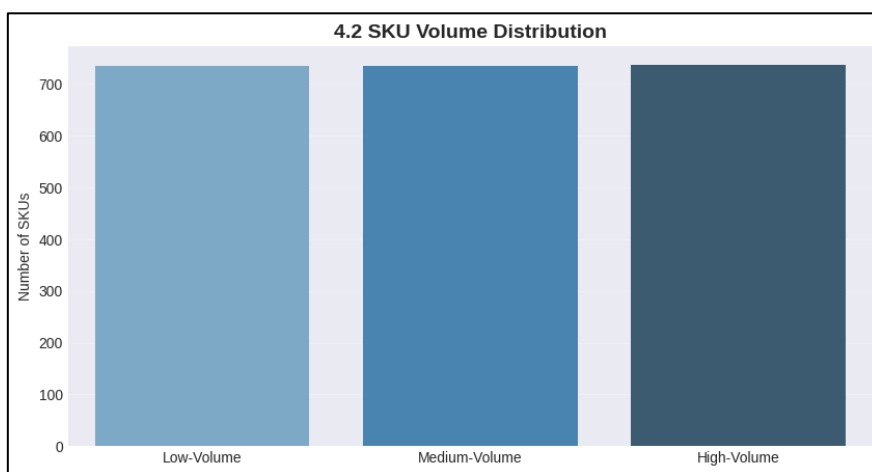


Fig.2: SKU Volume Distribution

4.3 Forecasting Performance

When it came to the actual forecasting, there was a massive gap between the basic methods and the smarter ones. A simple naive forecast ended up with a test RMSE of 232.04. It just couldn't keep up with the patterns. Ridge Regression, on the other hand, was much sharper, bringing that error

down to 24.00. Random Forest and XGBoost did okay, landing at 125.61 and 120.87, but they were still way behind Ridge. It seems those more complex models might have tripped over the gaps in the data or tried too hard to find patterns that weren't there. The takeaway here is that linear models, when given the right historical data to chew on, can be surprisingly tough to beat.

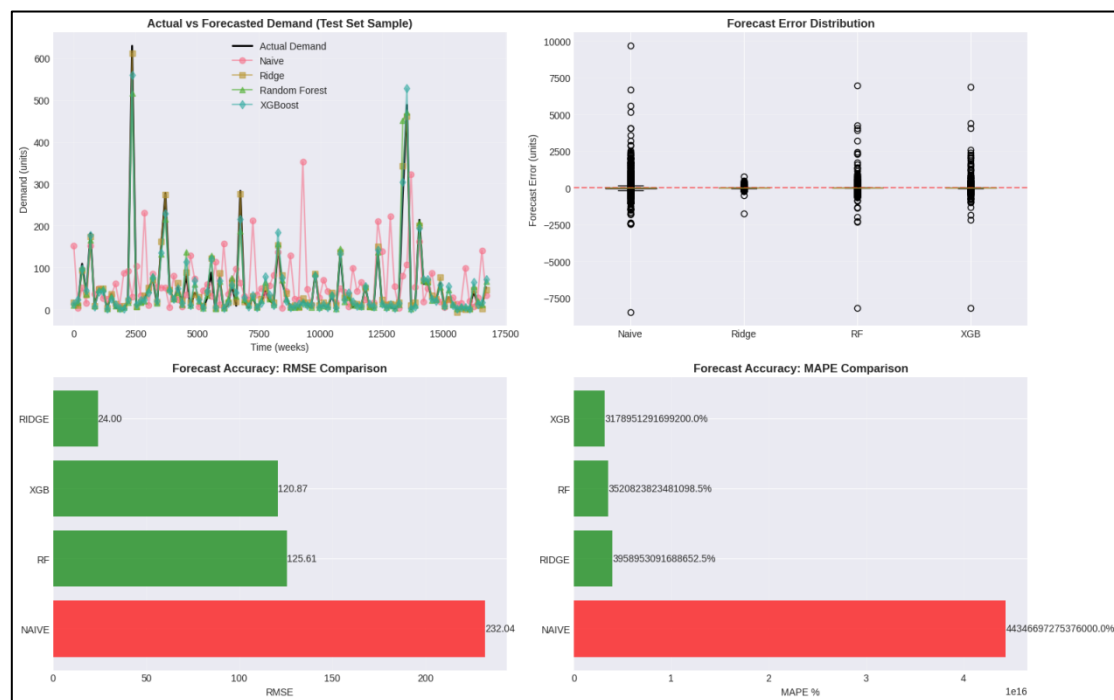


Fig.3: Forecast Accuracy Comparison

4.4 Dynamic Reorder Point Behavior

Watching how the reorder points changed was revealing. The versions driven by machine learning stayed much lower on average than the old-school static ones. This suggests a leaner, more responsive way of keeping stock. However, these reorder points jumped around a lot more. The standard

deviation was high, which shows the models were constantly reacting to what just happened in the market. While being adaptive is good, it does make things a bit more hectic. You get a system that reacts quickly, but the trade-off is a threshold that is always moving. The naive version was the opposite: very stable, but way too cautious and slow to change.

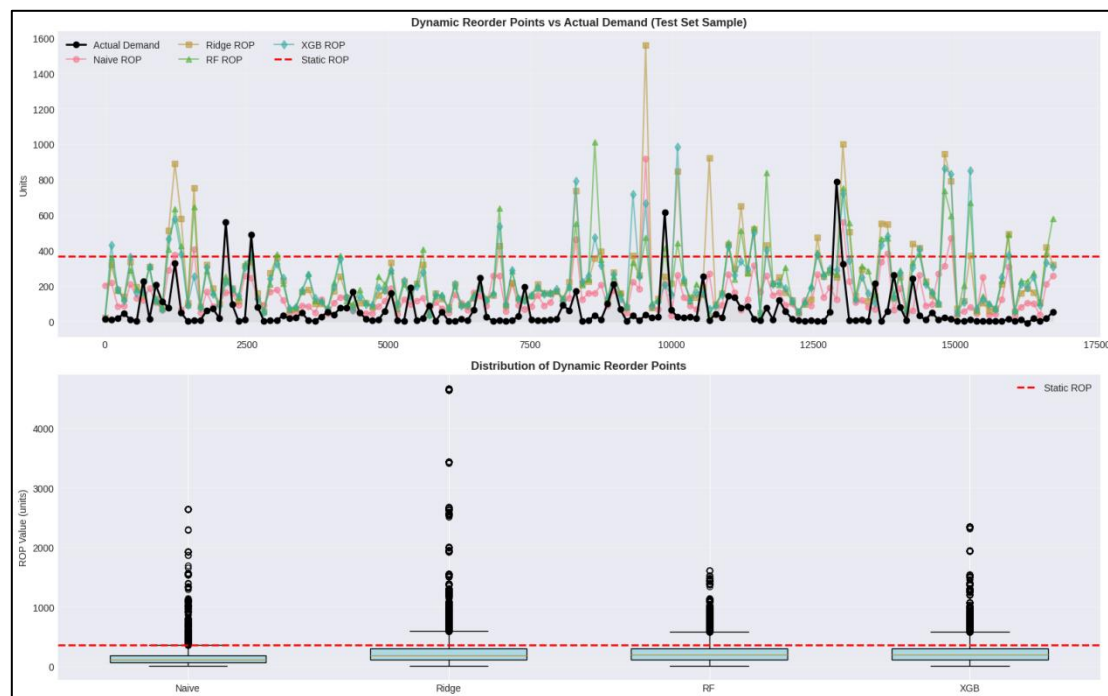


Fig.4: Dynamic Reorder Point Trajectories

4.5 Inventory Simulation Results

The simulation is where the rubber meets the road. Ridge Regression and the other models managed to hit fill rates over 70%, which is a big step up from the 54% managed by the naive approach. These dynamic reorder points did a better job of keeping products on the shelves when people actually wanted them. The catch, of course, is that

these models held more stock on average to make that happen. It is the classic balance: if you want better service, you usually have to pay for it with more inventory. Interestingly, the machine learning models also led to fewer orders overall. They seemed to bundle things better, which could make the actual process of ordering and receiving goods feel a bit more organized.

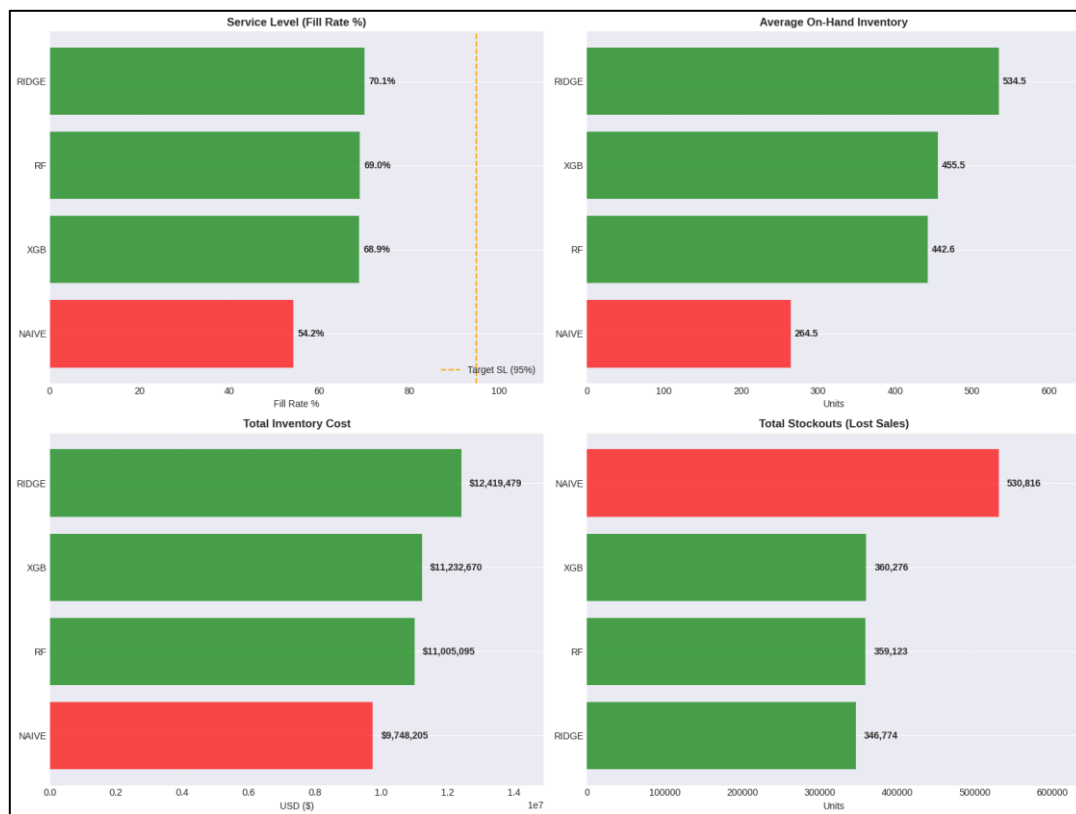


Fig.5: Inventory Performance Metrics

4.6 Sensitivity Analysis Results

The whole system is pretty sensitive to how long it takes for new stock to arrive. When the wait time went from one week to three, the fill rates dropped for everyone. That extra uncertainty is just hard to manage. Even so, the machine learning models still

beat the basic approach on service levels, even when things got tough. But because the naive model keeps less stock, it technically costs less in total. It really comes down to what a business values more: saving money on storage or making sure the customer actually gets their order.

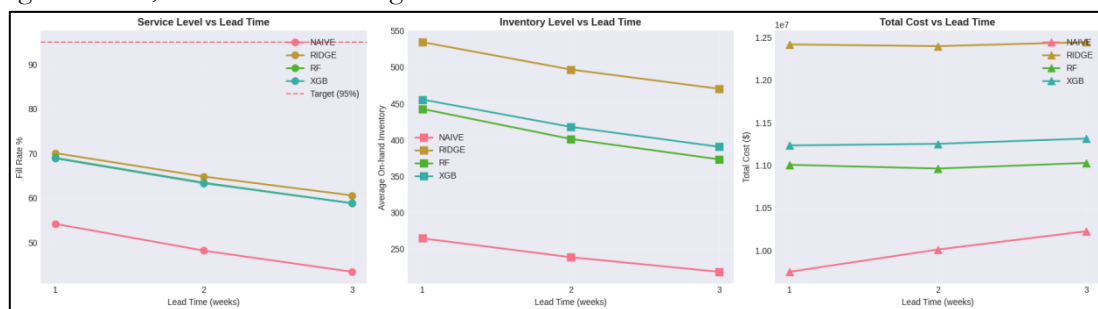


Fig.6: Sensitivity Analysis

4.7 Error Analysis

Looking at where the models failed is also useful. Everyone did fine when demand was low or steady. The wheels came off during those massive, unexpected spikes in demand. None of the models quite figured out how to predict those rare, huge surges, leading to stockouts. Returns also caused some headaches, as negative numbers can give the

models a confusing signal. It shows that while these tools are great, they still struggle with the weird outliers and the messy parts of retail, like returns and sudden crazes. In the end, these machine-learning policies are a clear step up in terms of being flexible and keeping service levels high, but they aren't a magic fix for every inventory problem.

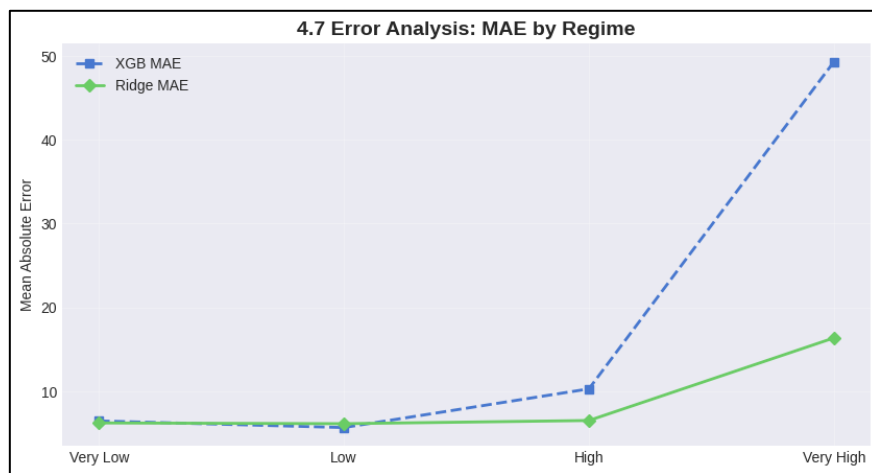


Fig.9: Error Analysis by Regime

5. DISCUSSION

5.1 Does AI Improve Reorder Point Decisions

The data shows pretty clearly that machine learning helps with reorder point decisions in a simulated setup, though it isn't a magic fix for everything. The biggest win shows up in service levels. It turns out that when a system actually learns demand patterns instead of just guessing, inventory levels and actual customer needs start to line up much better than they do with a basic baseline. That said, these improvements don't hit every metric the same way. Even though forecasting gets a lot more accurate, the physical reality on the ground often involves carrying more stock. It seems the models deal with uncertainty by padding things out with bigger safety buffers. It's a classic trade-off where the math tries so hard to avoid being wrong that it leans toward being safe, which ends up pushing those decision thresholds higher. The takeaway here is that sticking AI into the mix doesn't just automatically result in a perfect reorder point. It mostly sharpens one part of the machine, demand estimation, and that ripples down into the final numbers. It matters because the success of a reorder point is tied to how well those predictions are calibrated. If a model thinks things are more volatile than they really are, the warehouse fills up too much. If it's too confident, things run out. Syntetos and Boylan (2005) pointed out a while back that getting those demand traits right, especially when sales are hit-or-miss, is the make-or-break factor for inventory performance [27].

Basically, better data is just the starting line, not the finish.

Looking at how predictive tech works in other fields helps put this in perspective. For example, Rahman et al. (n.d.) found that machine learning can spot financial crisis patterns that old-school methods miss, but the real value is still tied to how humans or systems actually use those signals [22]. Reza et al. (2025) noted something similar: the systems are great at catching early warning signs, yet the impact is only as good as the final decision it triggers [24]. Then there is the issue of the signal-to-noise ratio, which Jakir (2025) emphasizes as a deciding factor in whether these tech upgrades actually lead to better results [17]. In this specific study, the models are definitely better at pulling a clear signal out of messy demand data, but turning that signal into a specific reorder number requires a careful hand to keep things both steady and fast.

5.2 Forecast Accuracy vs Inventory Performance

One of the big takeaways is that just because a forecast is more accurate doesn't mean the inventory management becomes better in a straight line. The machine learning models cut down on errors significantly, yet the resulting policies still end up wrestling with the classic tug-of-war between keeping customers happy and keeping costs down. Inventory is a bit lopsided by nature. Guessing too low means empty shelves and lost money, while guessing too high means paying to store stuff that isn't moving. A model might look great on paper because it has a low average error,

but it can still mess up inventory if it consistently misses the mark during the most important weeks.

This isn't exactly a brand-new concept. Back in 2012, Babai et al. showed that being better at predicting the future doesn't always lead to a better-run warehouse, especially if the policy doesn't account for how long it takes to get new stock or how much demand jumps around [4]. This study backs that up. The ML models hit high service levels, but do so by holding more inventory. Rahman (2025) noted that systems designed to find tiny patterns might be good at "seeing" demand, but they can create a new kind of mess if the thresholds for taking action aren't tied to the actual business goals [23]. In this specific case, the ML reorder points react more sharply to what's happening, which stops stockouts but keeps more boxes sitting on the shelves. This trade-off is a reminder that models need to be judged on how they work in the real world, not just how well they do on a math test. A model isn't truly working if it only gets the forecast right but doesn't actually make the business run smoother. The results here suggest that these dynamic reorder points need to be tweaked based on what matters more, never running out or saving on storage, instead of just assuming a "better" forecast solves everything.

5.3 Operational Implications

The results here offer a pretty clear roadmap for how to actually build these systems. The way this experiment was set up shows that it is possible to take daily transaction data and use it to drive reorder decisions. In a real shop, this would look like a cycle: new data comes in, the computer updates its guess for every single item, and the reorder points move up or down based on the latest trends. This moves things away from old-school, "set it and forget it" rules and toward a system that actually learns and changes as things happen. From a day-to-day perspective, using these moving reorder points makes a business much faster at reacting to weird spikes in demand, which keeps customers from leaving empty-handed. However, since those points are always moving, someone has to keep an eye on them so the ordering doesn't get out of control. This points toward a "hybrid" way of working. The AI does

the heavy lifting with the data, but humans stay in the loop to make sure things stay grounded, especially for products that are hard to predict. Waller and Fawcett (2013) argued that data-heavy supply chains work best when the tech is baked into the decision process rather than just left to run on its own [28]. The AI provides the suggestion, but it's still just a tool.

This idea works for more than just retail. Similar setups are used in big factories to figure out when a machine might break. In those cases, the best results come from mixing data insights with the reality of how the factory actually runs. Alam et al. (2026) showed that deep learning can be very accurate in industrial settings, but it only works if those predictions fit into the way the work gets done [2]. It's the same story here. The real power of machine learning in inventory isn't just the "prediction" itself; it's how that prediction is woven into a system that actually understands the goal of the business.

5.4 Sensitive Results Interpretation

It is important to look at these results through the lens of how the experiment was actually built. This study used a simulated inventory system based on transaction data, so it isn't a perfect mirror of how the retailer actually runs their day-to-day operations. It doesn't prove exactly how things would go if this were launched in a real warehouse tomorrow. The findings show how different reorder point rules stack up against each other under a specific set of rules, like fixed wait times for shipments and a setup where any missed sale is just gone forever. Keeping things in perspective helps prevent making claims that are too broad. A machine learning model might look great in a computer simulation, but the messy reality of a supply chain involves things this data doesn't see. Suppliers might be late, nobody might know exactly how much stock is on the shelf, or a promotion might suddenly spike demand. Even things like customer returns or weird one-off transactions can mess up the demand numbers. These gaps are worth noting so the conclusions stay realistic and tied to what was actually tested.

Even with those hurdles, using a simulation makes sense when there isn't a full set of live operational

records to work with. It creates a level playing field where different methods can be tested under the same conditions. This is a common way to do things in predictive research, where you test a framework in a sandbox before trying it out for real. The main point here isn't to say one way is definitely the best ever, but to provide a solid look at how moving reorder points around can help guide future work and actual business use.

5.5 Practical Benefits

Even with the constraints mentioned, there are clear perks to using machine learning for shifting reorder points. The big win is cutting down on stockouts. When products are actually on the shelf when people want them, customers stay happy. Because this system adjusts to what people have been buying lately, it gets better at seeing shifts coming and keeping service levels where they need to be. It gives a company better control over its goals, letting it hit specific targets instead of just guessing with the same old static numbers. There is also the benefit of finally seeing what is actually happening with demand at the individual product level. Building these models requires looking closely at what drives sales, like time of year or price changes. This kind of deep dive helps with more than just ordering stock; it can help figure out better pricing or what products to carry in the first place. These models are great at finding patterns that aren't obvious at first glance. It is similar to how researchers use data to find hidden networks in corporate finance (Dola et al., 2024) [10]. Looking at things through a graph-based lens in finance also shows how everything is connected, which is a logic that carries over well to supply chains (Islam et al., n.d.) [16].

Being able to spot weird or sudden changes is another plus. This kind of AI is already used to catch red flags in cybersecurity and banking, where finding the odd one out is a huge part of managing risk (Aashish et al., 2025) [1]. In a warehouse setting, this means the system can flag a sudden demand spike or a weird wave of returns that might need a human to step in. This same kind of pattern-spotting has been used to track shady money moves, proving that AI is quite good at untangling complex behaviors (Shawon et al., 2025)

[25]. Finally, putting AI into the mix helps make decisions more consistent and easier to track. In the legal world, AI has been used to help make sure complex decisions follow a logical path (Miah et al., 2026) [21]. For a supply chain, this means having a system that actually explains why it wants to order more of something. This helps managers trust the computer's suggestions. The real value here is combining smart data analysis with everyday tasks, creating a way to handle inventory that can grow along with the data. Hasan et al. (n.d.) further proposed an AI-powered framework for inventory optimization that links predictive analytics with stock management strategy, arguing that adaptive inventory systems can improve replenishment responsiveness, reduce operational inefficiencies, and strengthen coordination across supply chain activities [13].

6. LIMITATIONS

Every study has its hurdles, and being clear about them is the only way to make sense of the results. The biggest issue here comes down to the data itself. The UCI Online Retail II dataset is basically a huge list of sales transactions. It doesn't actually show what was happening in the warehouse. There's no record of stock-on-hand, when orders were placed with suppliers, how long deliveries took, or any actual purchase orders. Because that information is missing, it was impossible to reconstruct how the retailer actually managed their stock or to compare these models against real-world human decisions. The reorder points were built inside an experimental bubble instead of being pulled from history. This means the whole evaluation is looking at simulated choices, not real-life replenishment habits. That leads right into the next problem: the inventory simulation. Using a simulation is a smart way to test things out in a controlled space, but the answers it gives are only as good as the rules it follows. This one assumed lead times never changed, that every missed sale was gone forever, and that the ordering policy stayed the same. In the messy world of supply chains, lead times jump around, customers wait for backorders, and suppliers have their own quirks. These shortcuts were necessary to keep the math from getting out of hand, but it does mean the

results might not perfectly translate to a real warehouse floor.

The costs used in the math are another thing to keep in mind. Things like holding costs or the price of running out of stock were picked as set parameters. They weren't calculated from actual bank statements or company ledgers. This works great for comparing one method against another on a level playing field, but the final numbers should be seen as relative scores. They aren't exact dollar-for-dollar predictions of what a company would save. Along those lines, the starting inventory and the rules for restocking were kept the same for every single item, which ignores how different products usually need different treatment. Then there is the way items were picked for the study. Only products with a decent amount of sales activity made the cut. This creates a bit of a bias because it focuses on the stuff that is actually easy to predict. Items that only sell once in a blue moon are mostly left out, which might make the models look a bit more successful than they would be in a store full of slow-movers. Even the demand data isn't perfect. If a store runs out of something, the sales drop to zero, but the demand is still there, you just can't see it in the data. This missing "invisible" demand can make the forecasts and reorder points a bit lower than they should be during busy times.

Returns and cancellations make the picture even fuzzier. When a customer brings something back, it creates a negative number that messes with the demand signal. Even though the study tried to balance these out to find the net demand, it can still hide the real rhythm of how people buy things. Plus, there is no info on big outside factors. Marketing pushes, holiday promos, or even a global supply chain hiccup weren't in the dataset. Without those details, the models can't fully explain why demand suddenly spikes or dips. Lastly, all of this comes from one specific retail dataset. It's a great, detailed source, but it might not represent how things work in other industries. A pharmacy or an auto parts warehouse has different rules and stresses than an online gift shop. The findings here are a solid look at this specific experimental setup, but they aren't necessarily

universal truths that apply to every supply chain on the planet.

7. FUTURE WORK

These gaps in the research actually point toward some pretty interesting next steps. The most obvious move would be to find a dataset that actually includes the "hidden" warehouse stuff, real stock levels, when the truck arrived, and the actual purchase orders. Having that would let researchers test these reorder point ideas against real-world decisions. It would be a much tougher test. It would also be great to stop pretending lead times are fixed and instead model them as the unpredictable variables they really are. Another path involves getting more comfortable with uncertainty. Instead of the model just spitting out a single number for a forecast, it could give a range of possibilities. Using these probabilistic forecasts would be a much more natural way to figure out safety stock. It would help the reorder points adapt better when things get shaky. Comparing these machine learning models against old-school methods specifically built for weird, jittery demand, like Croston's method, would also show where the high-tech stuff actually wins and where it doesn't.

Future projects could also try being a bit more flexible with how different products are handled. Instead of one-size-fits-all, the rules for restocking could change based on how a product behaves, like how often it sells or where it is in its lifecycle. New versions of the study could also look at what happens when customers are willing to wait for backorders, which would give a much fuller picture of how inventory flows. Looking at the bigger picture, this could be scaled up to look at entire supply chain networks. Usually, a choice at a retail shop ripples up to the warehouse and then back to the factory. Studying those connections would be a lot closer to how the world actually works. On a more practical note, building this into a dashboard would be a huge win. A tool where someone could see the demand patterns, tweak the settings, and see the results in real-time would turn this from an academic exercise into something a manager could actually use. The final piece of the puzzle is just testing this on more than one store. Running the same logic across different industries and different

types of products would prove whether these machine learning tricks are truly robust. Doing that wouldn't just fix the issues found in this study, it would help build inventory systems that are actually ready for the real world.

Conclusion

This study set out to evaluate whether machine-learning-driven dynamic reorder point policies can improve inventory decision-making compared with traditional approaches. By constructing a comprehensive experimental pipeline that links demand forecasting, reorder point generation, and inventory simulation, the research provides a structured framework for assessing the operational value of predictive models in supply chain contexts. The results confirm that machine learning models are capable of capturing complex demand patterns and translating these insights into more responsive reorder thresholds. This leads to improved service levels and reduced stockouts, demonstrating the potential of AI-driven approaches to enhance inventory management.

However, the findings also highlight that improved forecasting accuracy does not automatically guarantee optimal inventory outcomes. The observed increase in inventory levels under machine-learning-based policies reflects a

fundamental trade-off between service quality and cost efficiency. This underscores the importance of evaluating inventory systems using both predictive and operational metrics, rather than relying solely on forecasting performance. The study further shows that model effectiveness varies across demand regimes and is sensitive to key assumptions such as lead time, indicating that dynamic reorder point policies must be carefully calibrated to the specific characteristics of the supply chain environment.

A key contribution of this research is its emphasis on integration. Rather than treating forecasting and inventory control as separate problems, the study demonstrates the value of connecting these components within a unified decision-support framework. This approach enables a more realistic assessment of how predictive models influence operational outcomes and provides actionable insights for practitioners. At the same time, the study acknowledges the limitations of simulation-based evaluation and the need for validation using real inventory data. In conclusion, AI-driven dynamic reorder point strategies represent a promising direction for modern inventory management, but their effectiveness depends on thoughtful design, careful calibration, and alignment with operational objectives.

References

- Aashish, K. C., Zamil, M. Z. H., Mridul, M. S. I., Akter, L., Sharmin, F., Ayon, E. H., ... & Malla, S. (2025). Towards eco-friendly cybersecurity: Machine learning-based anomaly detection with carbon and energy metrics. *International Journal of Applied Mathematics*, 38(9s).
- Alam, M., Shil, S. K., Sharmin, F., KC, A., Md, A. H., Ali, M., ... & Malla, S. (2026). Hybrid deep learning models for equipment failure prediction in US industrial systems. *International Journal of Applied Mathematics*, 39(1s).
- Axsäter, S. (2006). *Inventory control* (2nd ed.). Springer.
- Babai, M. Z., Ali, M. M., Nikolopoulos, K., & Boylan, J. E. (2012). Forecasting and inventory performance in a two-stage supply chain. *International Journal of Production Economics*, 135(1), 463–473.
- Banks, J., Carson, J. S., Nelson, B. L., & Nicol, D. M. (2010). *Discrete-event system simulation* (5th ed.). Pearson.
- Bertsimas, D., & Thiele, A. (2006). A robust optimization approach to inventory theory. *Operations Research*, 54(1), 150–168.
- Bhowmik, P. K., Subha, D. T., Rahim, A., Mohammed, A. A., Begum, M., Chowdhury, R., ... & Shati, M. A. (n.d.). Self-adaptive machine learning models for financial risk forecasting: Handling non-stationarity in banking and cryptocurrency time series.
- Carbonneau, R., Laframboise, K., & Vahidov, R. (2008). Application of machine learning techniques for supply chain demand forecasting. *European Journal of Operational Research*, 184(3), 1140–1154.

- Chopra, S., & Meindl, P. (2016). *Supply chain management: Strategy, planning, and operation* (6th ed.). Pearson.
- Dola, A., Begum, S., Antara, U. K., Islam, M. R., Sultana, T., & Zabin, N. (2024). Machine learning models for detecting hidden collusion networks in US corporate finance. *Journal of Economics, Finance and Accounting Studies*, 6(1), 143–154.
- Fildes, R., Goodwin, P., Lawrence, M., & Nikolopoulos, K. (2008). Effective forecasting and judgmental adjustments. *International Journal of Forecasting*, 24(1), 3–23.
- Hadley, G., & Whitin, T. M. (1963). *Analysis of inventory systems*. Prentice-Hall.
- Hasan, M. R., Sarker, B., Mishu, K. P., Rahman, M. A., Shawon, R. E. R., & Debnath, P. AI-powered inventory optimization models: A strategic framework for improving stock management in our supply chains.
- Hyndman, R. J., & Athanasopoulos, G. (2018). *Forecasting: Principles and practice* (2nd ed.). OTexts.
- Islam, M. R., Subha, D. T., Pramanik, M. T., Akter, M., Sweet, M. M. R., Robbani, M. S., ... & Zeeshan, M. A. F. (n.d.). AI-driven decision support systems for optimizing working capital and customer experience in the US.
- Islam, M. Z., Sumsuzoha, M., Islam, M. R., Kawsar, M., Mithu, M. F. H., Pant, S., ... & Al Helal, M. A. (n.d.). Graph neural networks for systemic financial risk forecasting.
- Jakir, T. (2025). Signal-to-noise analysis of crisis indicators in global finance using artificial intelligence. *International Journal of Applied Mathematics*, 38(10s), 1815–1836.
- Kourentzes, N., Petropoulos, F., & Trapero, J. R. (2020). Improving forecasting by estimating time series structural components. *International Journal of Forecasting*, 36(4), 1389–1404.
- Law, A. M. (2014). *Simulation modeling and analysis* (5th ed.). McGraw-Hill.
- Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2018). The M4 competition. *International Journal of Forecasting*, 34(4), 802–808.
- Miah, M. N. I., Uddin, M. J., & Kakumani, M. (2026). Artificial intelligence in sentencing. *Frontiers in Computer Science and Artificial Intelligence*, 5(4), 30–43.
- Rahman, M. K., Hossain, M. S., Haque, S. U., Jahed, K. A., Robbani, M. S., Shati, M. A., ... & Pramanik, M. T. (n.d.). Machine learning models for early warning of financial crises.
- Rahman, M. S. (2025). Machine learning-enabled early warning system for detecting micro-inflation clusters. *International Journal of Applied Mathematics*, 38(12s), 2743–2769.
- Reza, S. A., Rahman, M. K., Rahman, M. D., Sharmin, S., Mithu, M. F. H., Hasnain, K. N., ... & Kabir, R. (2025). Machine learning enabled early warning system for financial distress. *arXiv preprint arXiv:2510.22287*.
- Shawon, R. E. R., et al. (2025). Detecting illicit cross-chain fund movement. *International Journal of Applied Mathematics*, 38(12s).
- Silver, E. A., Pyke, D. F., & Thomas, D. J. (1998). *Inventory management and production planning and scheduling* (3rd ed.). Wiley.
- Syntetos, A. A., & Boylan, J. E. (2005). The accuracy of intermittent demand estimates. *International Journal of Forecasting*, 21(2), 303–314.
- Waller, M. A., & Fawcett, S. E. (2013). Data science and predictive analytics in supply chains. *Journal of Business Logistics*, 34(2), 77–84.